Introducción a la Bioinformática:

Comparative Genomics: Sequence Alignments I

Luis Garreta

Doctorado en Ingeniería
Pontificia Universidad Javeriana – Cali

March 19, 2018

Motivation:
Evolutionary History of the Sequences

Any alignment between two or more nucleotide or amino acid sequences **represents an explicit hypothesis** regarding the evolutionary history of those sequences.

Motivation:

Comparisons of Sequences facilitate their Understanding

Comparisons of related protein and nucleotide sequences have facilitated advances in understanding the content and function of genetic sequences.

Motivation:
Solving Key Problems in Bioinformatics

Sequence alignments provide important information for solving many of the key problems in bioinformatics including:

- Find evolutionary relationships between organisms (genes, proteins), and
- Identify the function of a newly discovered genetic sequence;
- Predicting the structure and function of proteins.

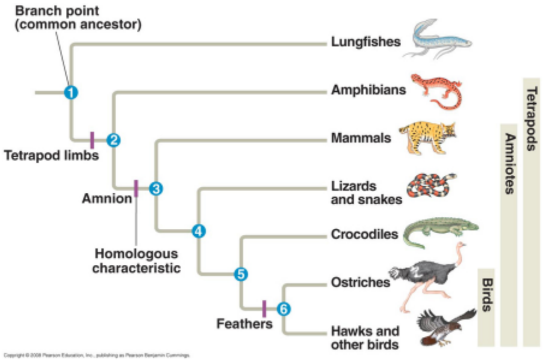
The Biological Problem

Basic Question in Biology

What properties are shared among organisms?

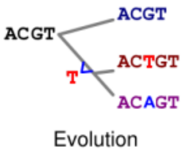
Homology: Organisms share Characteristics

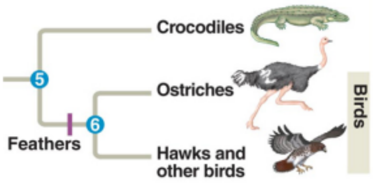
Descent from a common ancestor



http://www.bio.miami.edu/dana/160/160S13_5.html

Homology: Sequences match Positions





Sequence Similarity

Intuitively, similarity of two sequences refers to the degree of match between corresponding positions in sequence

Similarity between sequences

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

.

Similarity between strings

```
G A R F I E L D T H E L A S T F A - T C A T
| | | | | | | | | | | | | | | | |   | | | |
G A R F I E L D T H E V E R Y F A S T C A T
```

Similarity vs. Homology

- Similarity **does not imply** homology
- Similarity can occur by chance
- But, homology **is expected to cause** similarity

Homology and Evolution

Homology is more difficult to detect over greater evolutionary

```
#mutations
0:             agtgtccgttaagtgcgttc
8:             agtgtccgcttcaaggggcgt
64:            acagtccgttcgggctattg
256:           cacgagtaagatatagct
1024:          acccttatctacttcctggagtt
2048:          agcgacctgcccaa
4096:          caaac
```

Sequence Alignment

Alignment specifies which positions in two sequences **match**

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

Edit Operations

Different types of possible mutations:

- Match: Points where a single base do not change
- Mistmatch: substitution (point mutation) of a single base
- Indel: insertion or deletion of a base with respect to the ancestor sequence:
- Gap**:** Result of an insertion or deletion in the sequence

```
G G A T C G - - G A T T C G A A T G A T T C
| | | | | |     | | | |       | | | | | | |
G G A T C G C C T G C C - - - A T G A T T C
```

- **13 matches, 4 mistmatches**: 5 indels (2 insertions (🟨) , 3 deletions (🟩))

Sequence Alignment Questions

- What sorts of alignments should be considered?
- How to score alignments?
- How to find optimal or good scoring alignments?
- How to evaluate the statistical significance of scores?

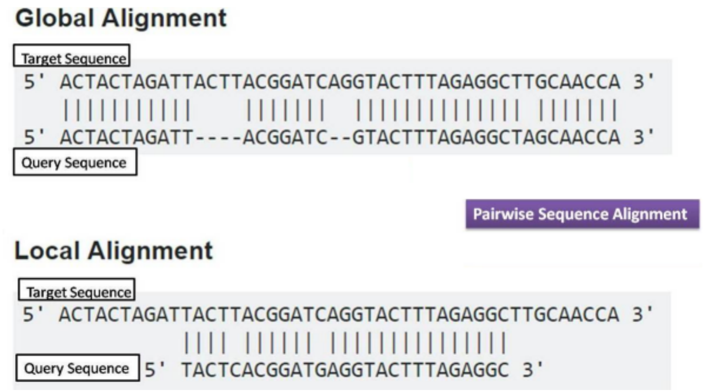
First Question:

What sorts of alignments should be considered?

Types of Alignments

- Pairwise Alignments: Between two sequences
  - Global Alignments
  - Local Alignments
- Multiple Alignments: Between more than two sequences

Pairwise Alignments:



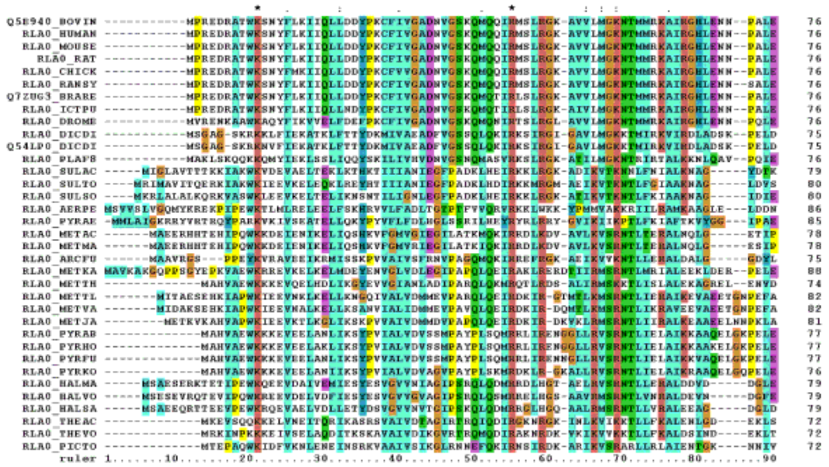
Global Alignment

```
Target Sequence
5'  ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA  3'
    |||||||||||    |||||||  |||||||||||||| |||||||
5'  ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA  3'
Query Sequence
```

Local Alignment

```
Target Sequence
5'  ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA  3'
              |||| |||||| |||||||||||||||
Query Sequence  5'  TACTCACGGATGAGGTACTTTAGAGGC  3'
```

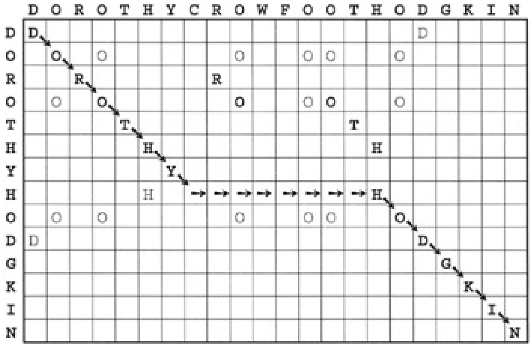
Multiple Alignments



```
                                 *   .      :       .          .   *         : : :      .
Q5E940_BOVIN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_HUMAN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_MOUSE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
    RLA0_RAT -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_CHICK -----------MPREDRATWKSNYFMKIIQLLDDYPKCFVVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_RANSY -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--SALE   76
Q7ZUG3_BRARE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQTIRLSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_ICTPU -----------MPREDRATWKSNYFLKIIQLLNDYPKCFIVGADNVGSKQMQTIRLSLRGK-AIVLMGKNTMMRKAIRGHLENN--PALE   76
  RLA0_DROME -----------MVRENKAAWKAQYFIKVVELFDEFPKCFIVGADNVGSKQMQNIRTSLRGL-AVVLMGKNTMMRKAIRGHLENN--PQLE   76
  RLA0_DICDI -----------MSGAG-SKRKKLFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIRDLADSK--PELD   75
Q54LP0_DICDI -----------MSGAG-SKRKNVFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIHDLADSK--PELD   75
  RLA0_PLAF8 -----------MAKLSKQQKKQMYIEKLSSLIQQYSKILIVHVDNVGSNQMASVRKSLRGK-ATILMGKNTRIRTALKKNLQAV--PQIE   76
  RLA0_SULAC -----MIGLAVTTTKKIAKWKVDEVAELTEKLKTHKTIIIANIEGFPADKLHEIRKKLRGK-ADIKVTKNNLFNIALKNAG-----YDTK   79
  RLA0_SULTO ----MRIMAVITQERKIAKWKIEEVKELEQKLREYHTIIIANIEGFPADKLHDIRKKMRGM-AEIKVTKNTLFGIAAKNAG-----LDVS   80
  RLA0_SULSO ----MKRLALALKQRKVASWKLEEVKELTELIKNSNTILIGNLEGFPADKLHEIRKKLRGK-ATIKVTKNTLFKIAAKNAG-----IDIE   80
  RLA0_AERPE MSVVSLVGQMYKREKPIPEWKTLMLRELEELFSKHRVVLFADLTGTPTFVVQRVRKKLWKK-YPMMVAKKRIILRAMKAAGLE---LDDN   86
  RLA0_PYRAE -MMLAIGKRRYVRTRQYPARKVKIVSKATELLQKYPYVFLFDLHGLSSRILHEYRYRLRRY-GVIKIIKPTLFKIAFTKVYGG---IPAE   85
  RLA0_METAC ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVGIEGILATKMQKIRRDLKDV-AVLKVSRNTLTERALNQLG-----ETIP   78
  RLA0_METMA ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVRIEGILATKIQKIRRDLKDV-AVLKVSRNTLIERALNQLG-----ESIP   78
  RLA0_ARCFU ------MAAVRGS---PPEYKVRAVEEIKRMISSKPVVAIVSFRNVPAGQMQKIRREFRGK-AEIKVVKNTLLERALDALG-----GDYL   75
  RLA0_METKA MAVKAKGQPPSGYEPKVAEWKRREVKELKELMDEYENVGLVDLEGIPAPQLQEIRAKLRERDTIIRMSRNTLMRIALEEKLDER--PELE   88
  RLA0_METTH -------------MAHVAEWKKKEVQELHDLIKGYEVVGIANLADIPARQLQKMRQTLRDS-ALIRMSKKTLISLALEKAGREL--ENVD   74
  RLA0_METTL -------MITAESEHKIAPWKIEEVNKLKELLKNGQIVALVDMMEVPARQLQEIRDKIR-GTMTLKMSRNTLIERAIKEVAEETGNPEFA   82
  RLA0_METVA -------MIDAKSEHKIAPWKIEEVNALKELLKSANVIALIDMMEVPAVQLQEIRDKIR-DQMTLKMSRNTLIKRAVEEVAEETGNPEFA   82
  RLA0_METJA --------METKVKAHVAPWKIEEVKTLKGLIKSKPVVAIVDMMDVPAPQLQEIRDKIR-DKVKLRMSRNTLIIRALKEAAEELNNPKLA   81
  RLA0_PYRAB -------------MAHVAEWKKKEVEELANLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAQELGKPELE   77
  RLA0_PYRHO -------------MAHVAEWKKKEVEELAKLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAKELGKPELE   77
  RLA0_PYRFU -------------MAHVAEWKKKEVEELANLIKSYPVVALVDVSSMPAYPLSQMRRLIRENNGLLRVSRNTLIELAIKKVAQELGKPELE   77
  RLA0_PYRKO -------------MAHVAEWKKKEVEELANIIKSYPVIALVDVAGVPAYPLSKMRDKLR-GKALLRVSRNTLIELAIKRAAQELGQPELE   76
  RLA0_HALMA -----MSAESERKTETIPEWKQEEVDAIVEMIESYESVGVVNIAGIPSRQLQDMRRDLHGT-AELRVSRNTLLERALDDVD-----DGLE   79
  RLA0_HALVO -----MSESEVRQTEVIPQWKREEVDELVDFIESYESVGVVGVAGIPSRQLQSMRRELHGS-AAVRMSRNTLVNRALDEVN-----DGFE   79
  RLA0_HALSA -----MSAEEQRTTEEVPEWKRQEVAELVDLLETYDSVGVVNVTGIPSKQLQDMRRGLHGQ-AALRMSRNTLLVRALEEAG-----DGLD   79
  RLA0_THEAC -------------MKEVSQQKKELVNEITQRIKASRSVAIVDTAGIRTRQIQDIRGKNRGK-INLKVIKKTLLFKALENLGD----EKLS   72
  RLA0_THEVO -------------MRKINPKKKEIVSELAQDITKSKAVAIVDIKGVRTRQMQDIRAKNRDK-VKIKVVKKTLLFKALDSIND----EKLT   72
  RLA0_PICTO -------------MTEPAQWKIDFVKNLENEINSRKVAAIVSIKGLRNNEFQKIRNSIRDK-ARIKVSRARLLRLAIENTGK----NNIV   72
       ruler 1.......10........20........30........40........50........60........70........80........90
```

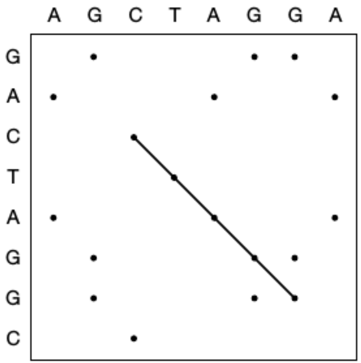
Dot Plot Matrix: Strings

String A: DOROTHYCROWFOOTHODGKIN
String B: DOROTHYHODGKIN



Dot Plot Matrix: Pair of Sequences

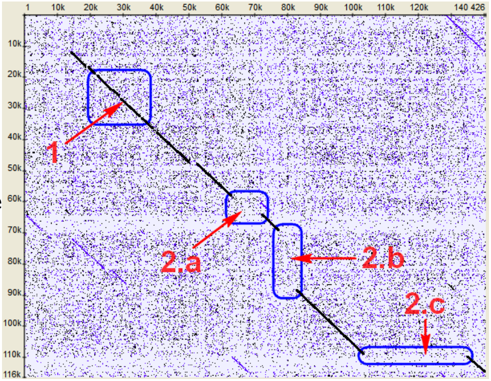
- Comparing two sequences:
  - AGCTAGGA
  - GACTAGGC
- Dots represent similarities between segments
- Diagonal of dots reveals similar elements



Not technically an ¨alignment¨ but it gives a picture of correspondence between pairs of sequences

Dot Plot Matrix: Interpretation

- **1 Matches:** looks like diagonals (continuous match or repeat)
- **2a Mutations:** gaps in the diagonal
- **2b Insertions:** gaps which lie only one axis (Y axis)
- **2c Deletions:** gaps which lie only one axis (X axis)



Dot Plot Matrix: Example

One alignment:

```
T C G G A T T C G T
| | | | | | | |
T C G C G T T C - -
```

Alternate alignment:

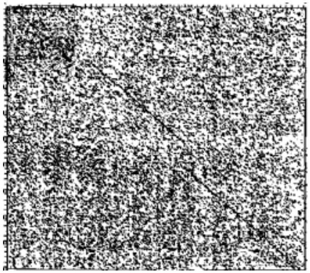
```
T C G G A T T C G T - -
| | |         | | |
T C G - - - - C G T T C
```

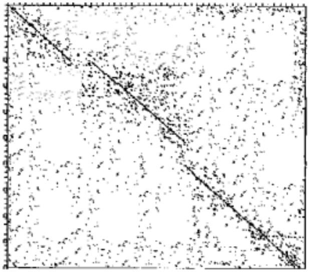
	T	C	G	G	A	T	T	C	G	T
T	●									
C		●								
G			●							
C								●		
G									●	
T						●				●
T							●			
C								●		

Dot Plot Limitations

Problems with larger sequences sharing extensive regions of similarity

- Solution: filtering using a window size and threshold





Alignment Considerations:

- **Homology is not a matter of degree**, individuals either share a common ancestor or they do not.
- **An lignment is simply a pairwise match** between the characters of each sequence.
- **A true alignment** (nucleotides or amino acids) reflects the evolutionary relationship between two or more homologous.
- **It needs a fractional value**, to decide if a true alignment reflects evolutionary relationship between two or more homologous

Second Question

How to score alignments?

Simple Alignments

Three possible **simple** alignments for AATCTATA y AAGATA:

```
AATCTATA                    AATCTATA                  AATCTATA
::   ::                      :                              :::
AAGATA                       AAGATA                      AAGATA
```

Three kinds of changes can occur:

1. **A mutation** replacing one character with another
2. **An insertion** adding one or more position
3. **A deletion** deleting one or more position

Scoring Simple Alignments

Scoring function for a Simple Alignment:

$$\sum_{i=1}^{n} \begin{cases} 1\text{: match score if seq1=seq2} \\ 0\text{: mismatch score if } seq1 \neq seq2 \end{cases}$$

Scoring the Alignments:

```
AATCTATA          AATCTATA          AATCTATA
::  ::             :                     :::
AAGATA             AAGATA             AAGATA
--------          --------          --------
Score = 4         Score = 1         Score = 3
```

Alignment with Gaps

- Insertions and deletions events complicates sequence alignments
- How many different possible subsets can be made from the larger set:
    - The number of possible alignments increase vastly.

$$C(7,2) = 28$$

Only 5 of the 28 possible alignments :

```
AATCTATA    AATCTATA    AATCTATA    AATCTATA    AATCTATA
AAG-AT-A    AA-G-ATA    AA--GATA    A-A-GATA    AA-GAT-A
```

Scoring Alignments with Gaps

Scoring function for a Simple Alignment:

$$\sum_{i=1}^{n} \begin{cases} -1 : \text{gap penalty, if seq1=''-'' or seq2=''-''} \\ +1 : \text{match score, if seq1=seq2} \\ 0 : \text{mismatch score, if } seq1 \neq seq2 \end{cases}$$

Scoring the Alignments:

```
AATCTATA   AATCTATA   AATCTATA   AATCTATA   AATCTATA
AAG-AT-A   AA-G-ATA   AA--GATA   A-A-GATA   AA-GAT-A
--------   --------   --------   --------   --------
Score =1   Score =3   Score =3   Score =2   Score =2
```

Origination and Length Penalties

- **Indel events (indels)**: Insertion and Deletion Events

What is more likely from an evolutionary perspective?

```
Without gaps          Multiple indels       Few indels
----------------      ----------------      ----------------
AATCTATAGGGTAGAT      AATCTATAGGGTAGAT      AATCTATAGGGTAGAT
  AAGATAGTAA          AA-G-AT-A-GT--AT      AAG--ATAG--TA--T
```

- Extended are more frequent than single multiple **indels events**
- Scoring function biased to reward alignments **extending gaps**

Scoring Alignments with Gap Penalty

Scoring function for a Simple Alignment:

$$\sum_{i=1}^{n}\begin{cases}-2 : \text{origination gap penalty, if seq1="-" or seq2="-"}\\ -1 : \text{length gap penalty, if seq1="-" or seq2="-"}\\ +1 : \text{match score, if seq1=seq2}\\ 0 : \text{mismatch score, if } seq1 \neq seq2\end{cases}$$

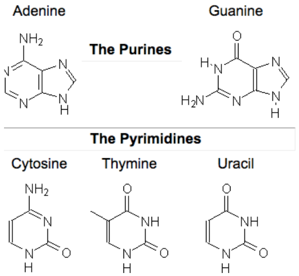
Scoring:

```
AATCTATAGGGTAGAT      AATCTATAGGGTAGAT
AA-G-AT-A-GT--AT      AAG--ATAG--TA--T
----------------      ----------------
Score = -3            Score = 0
```

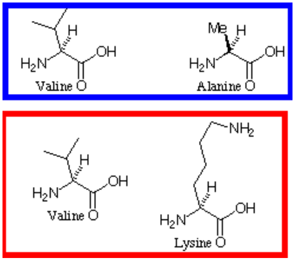
Scoring Matrices: Taking account conservative substitutions

Some substitutions are more common than others.

ADN, nucleotides:



Proteins, Amino acids:



Mismatch penalty can be broken down as gap penalty

Scoring Matrices: DNA Sequences

Identity Matrix

	A	T	C	G
A	1	0	0	0
T	0	1	0	0
C	0	0	1	0
G	0	0	0	1

BLAST Matrix

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

Transition Transv.

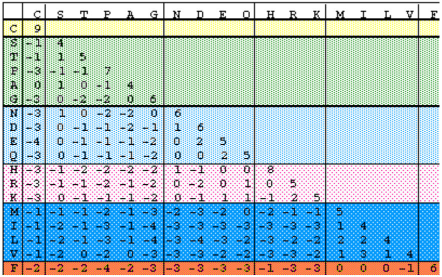
	A	T	C	G
A	5	-4	-4	**1**
T	-4	5	**1**	-4
C	-4	**1**	5	-4
G	**1**	-4	-4	5

- Scoring matrix is used to score each nongap position
  - Transitions transversion matrix provides mild penalty for transitions:
    - Purine (A or G) is replaced with another purine
    - Pyramidine (C or T) is replaced with another purine

Scoring Matrices: Amino Acid sequences

- PAM (Point Accepted Mutation):
  - Computed by observing substitution rates
  - Used to score closely related sequences
- BLOSUM (BLOcks SUbstitution Matrix):
  - Computed by clustering ungapped alignments
  - Used to score more distant related sequences

Blosum



	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F
C	9																	
S	-1	4																
T	-1	1	5															
P	-3	-1	-1	7														
A	0	1	0	-1	4													
G	-3	0	-2	-2	0	6												
N	-3	1	0	-2	-2	0	6											
D	-3	0	-1	-1	-2	-1	1	6										
E	-4	0	-1	-1	-1	-2	0	2	5									
Q	-3	0	-1	-1	-1	-2	0	0	2	5								
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8							
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5						
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5					
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5				
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4			
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4		
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4	
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6

Third Question

How to find optimal or good scoring alignments?

Types of Algorithms for Pairwise Alignments

- Exaustive search
- Recursive algorithm
- Dynamic programming

Exhaustive search

Idea: Search for each possible alignme

- It is not feasible for most sequences
  - Two modest-sized sequences of 100 and 95 nucleotides may produce ˜75 million possible alignments
  - For larger sequences, search becomes **intractable**
- Impossible to compute in a reasonable amount of time

Impossible to compute in a reasonble amount of time

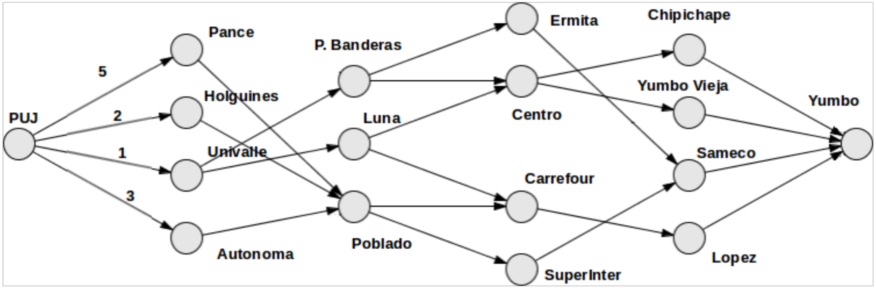
Dynamic Programming

- DP is a method for reducing a complex problem to a set of identical sub-problems.
- The best solution to one sub-problem is independent from the best solution to the other sub-problems.
- **DP is a bottom-up mechanism:** we solve all possible small problems and then combine them to obtain solutions for bigger problems.

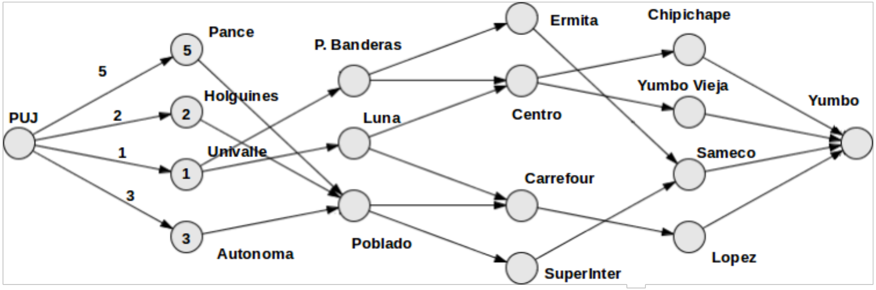
Example: the Fibonacci Series

- F(n) = F(n - 1) + F(n - 2)
- Using DP, we solve it subproblem once

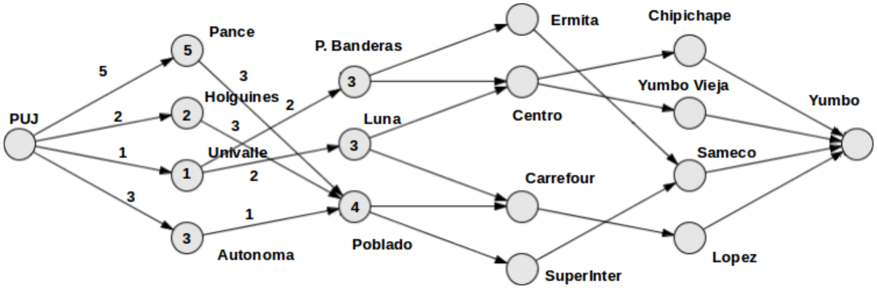
Example: Shortest Path Problem (Initial)



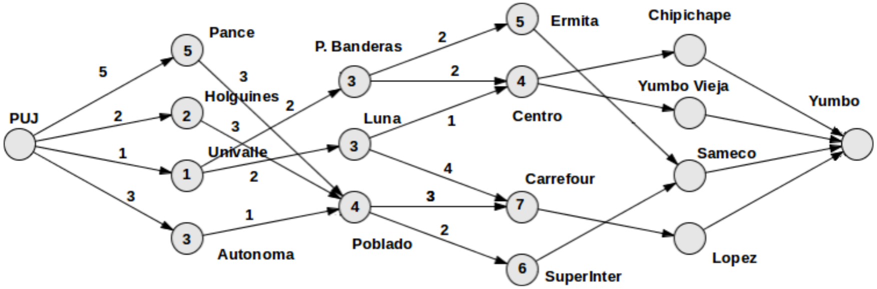
Example: Shortest Path Problem (01)



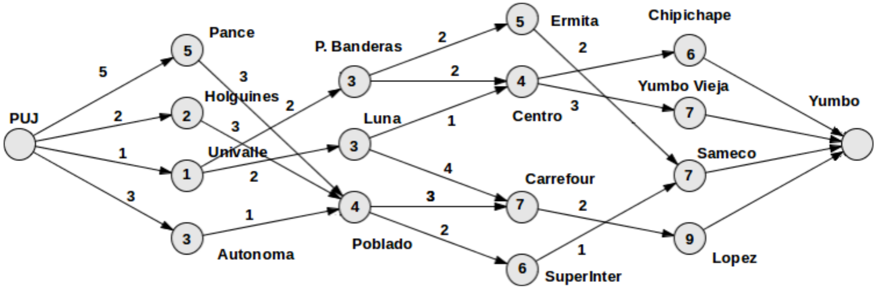
Example: Shortest Path Problem (02)



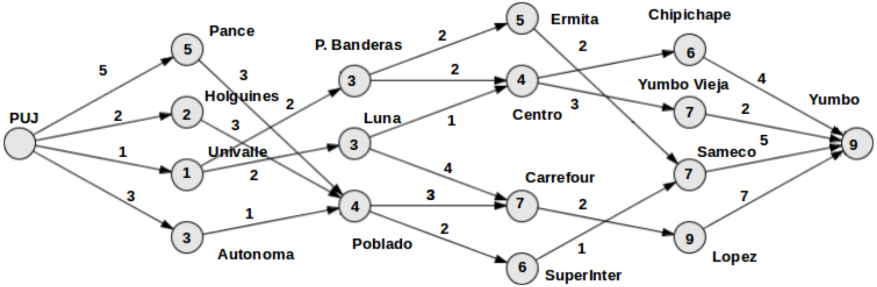
Example: Shortest Path Problem (03)



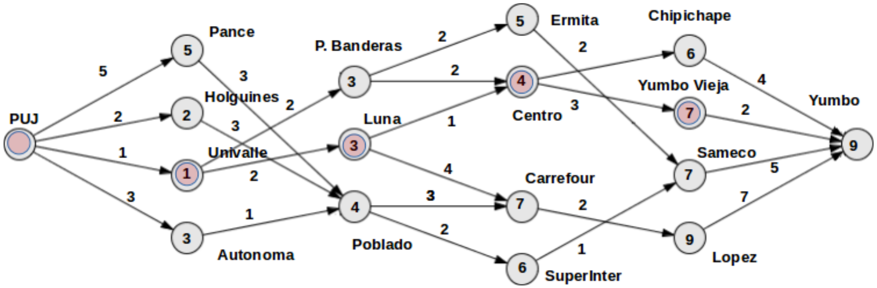
Example: Shortest Path Problem (04)



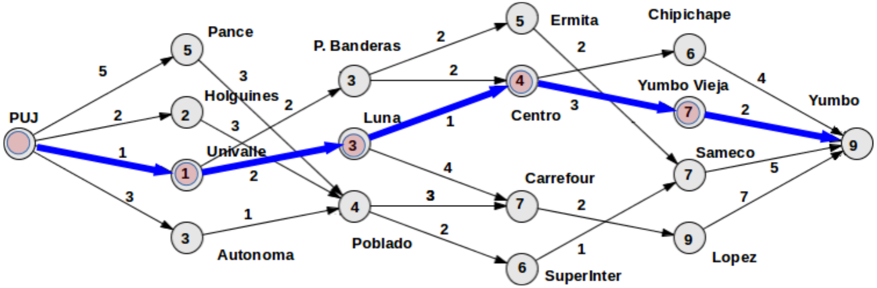
Example: Shortest Path Problem (Final)



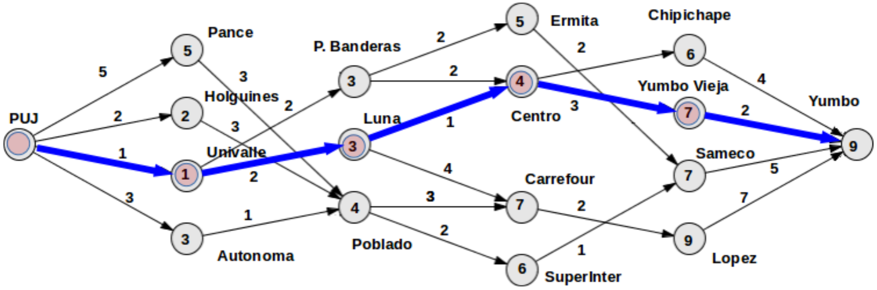
Example: Shortest Path Problem (Backtracking)



Example: Shortest Path Problem (Shortest Path)



Example: Shortest Path Problem (Recursive Algorithm)



```
ShortestPath (PUJ, Yumbo):
   min(  5 + ShortestPath (Pance, Yumbo);
         2 + ShortestPath (Holgines, Yumbo);
         1 + ShortestPath (Univalle, Yumbo);
         2 + ShortestPath (Autonoma, Yumbo)
       )
```

Too Many Recursive Calls

```
ShortestPath (Pance, Yumbo):
   min(  5 + ShortestPath (Poblado, Yumbo))
ShortestPath (Poblado, Yumbo):
   min(  5 + ShortestPath (Carrefour, Yumbo);
         5 + ShortestPath (SuperInter, Yumbo))
...
ShortestPath (Sameco, Yumbo):
   min(5)

5
```

```
ShortestPath (Univalle, Yumbo):
   min(  2 + ShortestPath (PBanderas, Yumbo);
         2 + ShortestPath (Luna, Yumbo))
ShortestPath (PBanderas, Yumbo):
   min(  2 + ShortestPath (Ermita, Yumbo);
         2 + ShortestPath (Centro, Yumbo);)
...
```

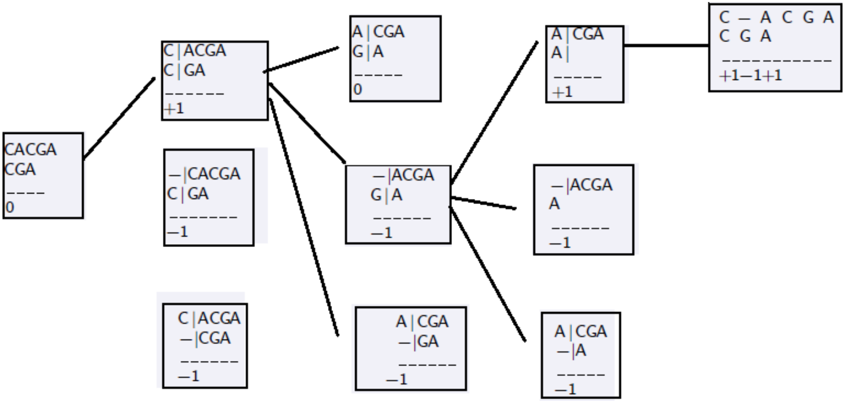
```
ShortestPath (Carrefour, Yumbo):
...
```

```
ShortestPath (SuperInter, Yumbo):
...
```

Needleman and Wunsch Algorithm
Global Alignments

- Needleman and Wunsch were the first to apply DP to sequence alignments
- Key to understanding DM approch to sequence alignment:
  Observing how the alignment problem is broken down into subproblems

Example: Align the sequences CACGA y CGA



Dynamic Programming Matrix

```
Sequence 1: CACGA
Sequence 2: CGA
```

Sequences CACGA y CGA

	–	C	A	C	G	A
–						
C						
G						
A						

Dynamic Programming Matrix: Initialization with Penalty Gaps

```
Sequence 1: CACGA
Sequence 2: CGA
```

- Uniform Penalty Gap of -1

Moves:

- Horizontal: gap in the X-Axis
- Vertical: gap in the Y-Axis
- Diagonal: match or mismatch

CACGA y CGA

	–	**C**	**A**	**C**	**G**	**A**
–	0	-1	-2	-3	-4	-5
C	-1					
G	-2					
A	-3					

Dynamic Programming Matrix:
Edit Operations and Scoring Function

Edit Operations

```
Sequence:                  CACGA
----------------------------
Substitution:              GACGA
Indel:
    Deletion  (Del):       -ACGA
    Insertion (Ins):       TGAGA
```

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

```
Sequence 1: CACGA
Sequence 2: CGA
```

CACGA y CGA

	–	C	A	C	G	A
–	0	-1	-2	-3	-4	-5
C	-1					
G	-2					
A	-3					

Dynamic Programming Matrix:
For each step

Scoring Function

```
Match:       +1
Mismach:      0
Indel:       -1
```

Compute the max score for each cell:

- According to the score
- According to the neighbors

Pos		1	2	3	4	5	6
		–	**C**	**A**	**C**	**G**	**A**
1	–	0	-1	-2	-3	-4	-5
2	**C**	-1	↘ ↓ +1 -1 → -1 **1**				
3	**G**	-2					
4	**A**	-3					

Dynamic Programming Matrix:
Finish with backtracking

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

$$\begin{array}{lcl} \texttt{CACGA} & \rightarrow & \texttt{CACGA} \\ \texttt{CGA} & \rightarrow & \texttt{C--GA} \end{array}$$

	–	C	A	C	G	A
–	0	-1	-2	-3	-4	-5
C	-1	↖**1**	←**0**	←**-1**	-2	-3
G	-2	0	1	0	↖**0**	-1
A	-3	-1	1	1	0	↖**1**

Dynamic Programming Matrix: Exercise

$$\begin{array}{lcl} \texttt{ACTCG} & \longrightarrow & ? \\ \texttt{ACAGTAG} & \longrightarrow & ? \end{array}$$

Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

	–	A	C	T	C	G
–						
A						
C						
A						
G						
T						
A						
G						

Dynamic Programming Matrix: Solution

```
ACTCG    →  AC--TCG
ACAGTAG  →  ACAGTAG
```

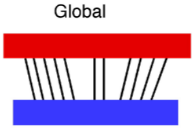
	–	**A**	**C**	**T**	**C**	**G**
–		-1	-2	-3	-4	-5
A	-1	1	0	-1	-2	-3
C	-2	0	2	1	0	-1
A	-3	-1	1	2	1	0
G	-4	-2	0	1	2	2
T	-5	-3	-1	1	1	2
A	-6	-4	-2	0	1	1
G	-7	-5	-3	-1	0	2

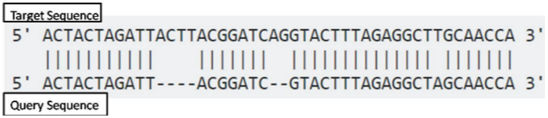
Scoring Function

```
Match:      +1
Mismach:     0
Indel:      -1
```

Considerations to the Needleman and Wunsch Algorithm

- The basic algorithm discussed so far implements a **global alignment**
  - It compares two sequences in their entirety




Target Sequence

```
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
   |||||||||||    |||||||  |||||||||||||| |||||||
5' ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA 3'
```
Query Sequence

- This is not always always the most desiderable way to align two sequences.

The best alignment for a short sequence

Example : AACACGTGTCT and ACGT

- From several possible alignments between AACACGTGTCT and ACGT
- The one we are most interested in is:

```
AACACGTGTCT
---ACGT----
```

- It is the most interesting because it demonstrates that the shorter sequence appears in its entirety within the longer sequence

Semiglobal Alignments

Avoid penalizing for gaps that appear at one or both ends of a sequence

```
AACACGTGTCT
---ACGT----
```

Initial gaps without penalties

- In the first sequence, initialize the first column of the table to all zeros
- In the second sequence, initialize the first row of the table to all zeros

End gaps without penalties

- In the first sequence, allow vertical moves without penalty in the last column of the table
- In the second sequence, allow horizontal moves without penalty in the last row of the table

Exercise

1. Study how to fill the partial score table for a semiglobal alignment
2. Construct the partial score table for the following two sequences using the semiglobal approach

```
AACACGTGTCT
ACGT
```

References

1. Fundamental Concepts of Bioinformatics (Chapter 2) by Dan E. Krane, Michael L. Raymer
2. Introduction to algorithms in bioinformatics (Chapter 3) by István Miklós